Obstacles to Linked Data

Simeon Warner

Metadata Working Group Forum, 2016-02-19

Credits

This talk is based in large part on Rob Sanderson's presentation **"RDF Failures and Linked Data Letdowns"** @ CNI 2013: Video & Slides.

I formed opinions about obstacles to linked data from working with the OAI-ORE team (Carl Lagoze, Herbert Van de Sompel, Pete Johnston, Michael Nelson and Rob Sanderson); the IIIF community (in particular Michael Appleby, Tom Crane, Rob Sanderson, Jon Stroop); and the LD4L team (at Cornell: Dean Krafft, Jon Corson-Rikert, Rebecca Younes, Lynette Rayle, Jim Blake, Muhammad Javed, Chiat Naun Chew, Jason Kovari and Steven Folsom; and others at Harvard and Stanford including, again, Rob Sanderson ;^).

Notwithstanding…

I think **Linked (Open) Data** is a good route, but we must be understand the obstacles to adoption in order to overcome them.

Not going to talk much about **Open** because we are trying to do that anyway, regardless of format and data model.

Some of the obstacles are because of the connectedness, openness and reusability we are trying to achieve, not specific to particular technologies that we might use to get there.

Linked data is not magic! But it has some nice affordances.

A Long Game

In 2013 Rob wrote:

- The Semantic Web was a *great* idea in 2003
- The Semantic Web is still a great *idea* in 2013

And now in 2016…

- We've pretty much dropped the term "Semantic Web"
- But Linked Data is still a great idea… and reaching prime time for *some applications*

Terminology - The Web

Assume terminology described in Architecture of the World Wide Web (AWWW) including:

- **resource** is a thing identified by a **URI** (or **IRI**)
- **dereference of a URI** is used a get a **representation** of the resource but may also provide information about related resources

---

Terminology - RDF

Assume working understanding of RDF – the Resource Description Framework – including:

- A **statement** or **triple** is comprised of three parts
  - **subject** - a URI naming a resource or a **blank node** (no URI)
  - **predicate** - a URI
  - **object** - a URI or a **literal**

RDF is the primary data type for Linked Data although others can be used.

Gotcha: even here we run into some complication because AWWW talks about resources as things identified by a URI, whereas the RDF spec talks about both URIs and literals as resources…

Topics

- Graphs
- Model Choices
- The Open World
- Ontologies and Identities
- User Interfaces
- Serializations
- Technologies
- Temporality

Graphs - Why Graphs?

Graphs are very powerful for modeling reality

- RDF is a particular graph model with directed, labeled edges (others possible)
- Extremely flexible
- Novel information can be automatically inferred
- Interesting questions can be asked based on structure
- Tree structures are just simple Graphs (often directed, acyclic, with known root node)

Graph Query

Graph querying is complicated

- Graph: *Structure* and *Data* important
  - … but data currently treated as second class citizen
- Other: Only *Data* important, so easier to work with
  - … and sophisticated data queries supported

Graph query to find books written by "Sanderson":

```
?book a b:Book ; dc:creator ?who .
?who a f:Person ; f:name "Sanderson"
```

Or when you don't need to worry about structure:

```
au:Sanderson
```

Graph Visualization

Visualization potentially powerful, but hard to get right

- Documents are easy to visualize (e.g. HTML browser, PDFs, images)
- Metadata records have clear boundaries (in or not-in)
- We are quite familiar with trees (e.g. files and directories)

- Data visualization understood

Graph visualization usually terrible but *good visualization very powerful*

- Often use simplified data and/or carefully arranged
- Reveal structure and/or links between structure and content

---

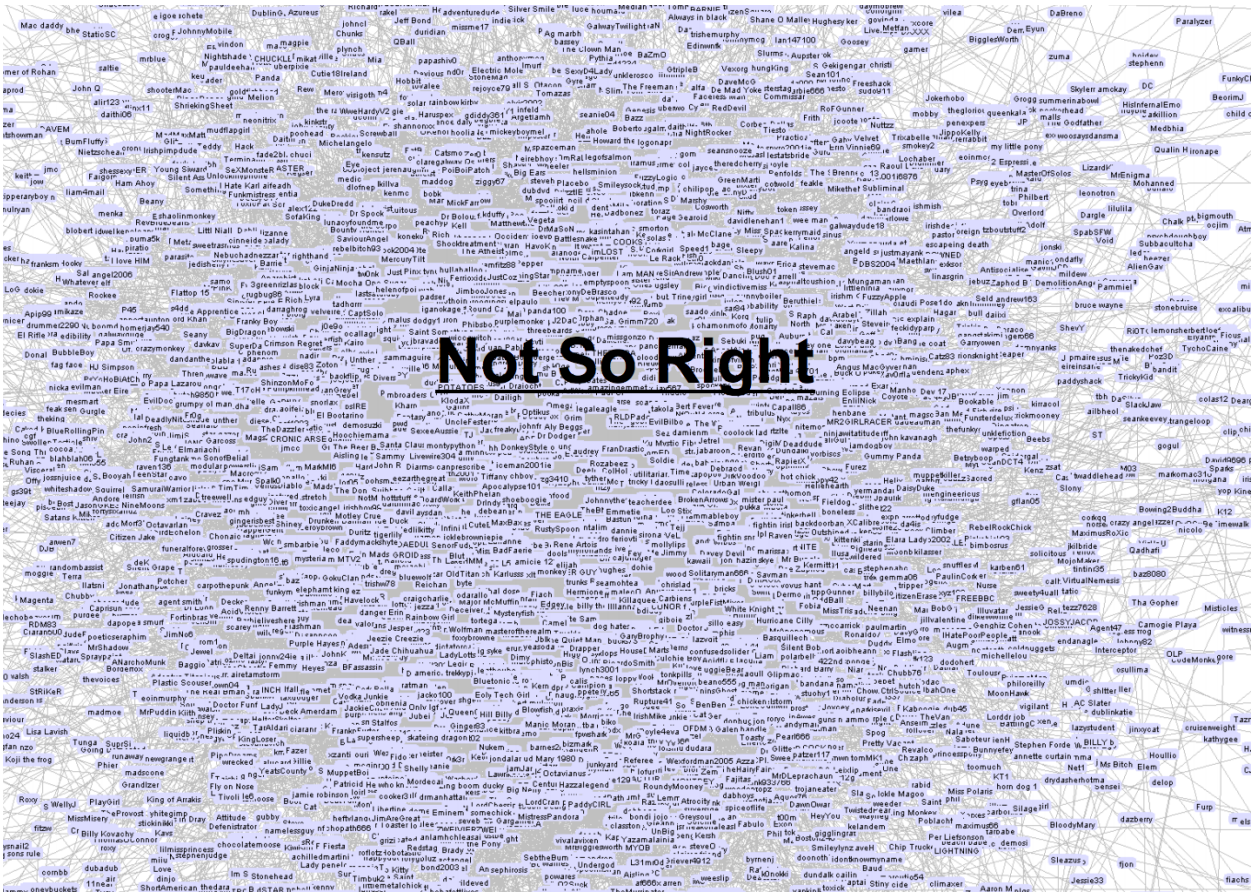
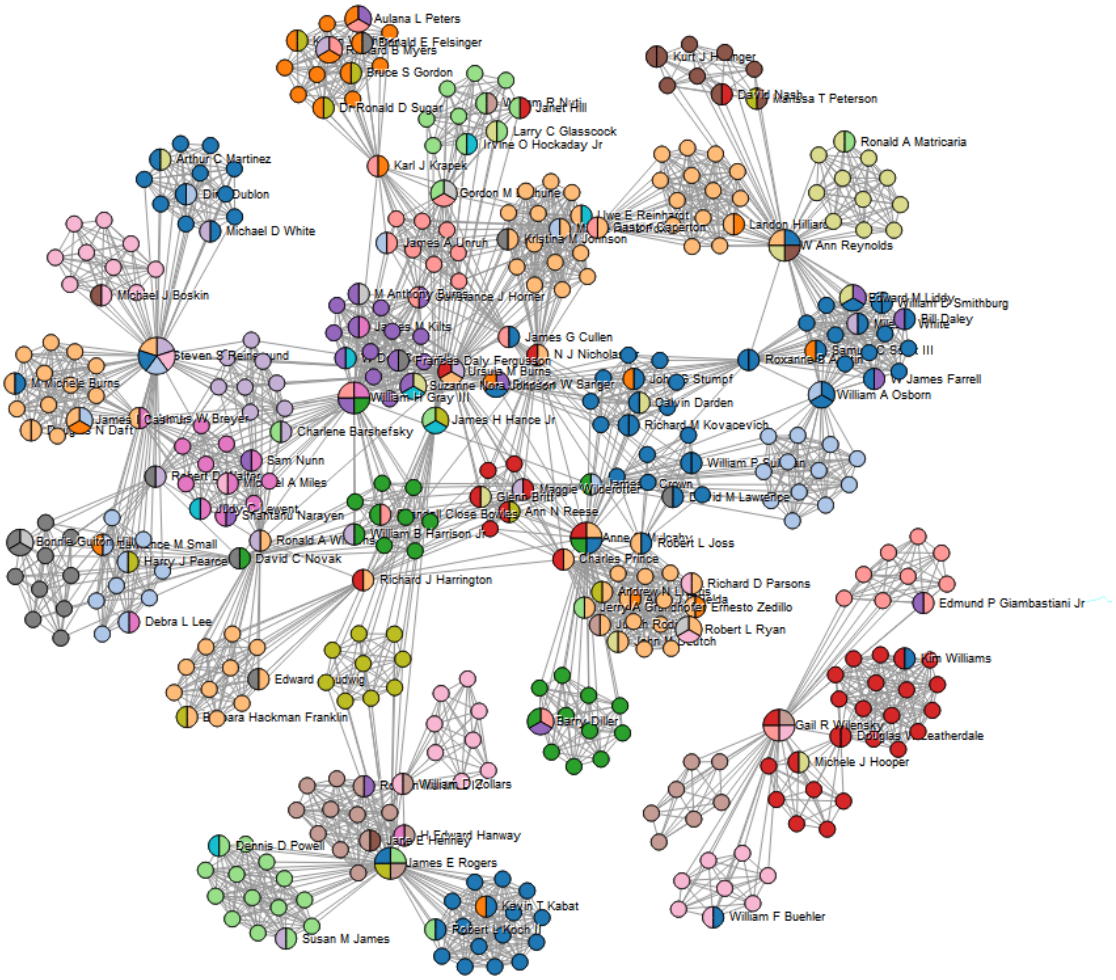


Figure 1: Unhelpful visualization

See description of Facebook data

(Structure alone often easier to show than combination with data)



See live version and description

Graphs - Mitigation

- Graph complexity is real and this expressiveness is reason for choosing them
- Use tools to handle/hide complexity (e.g. triplestores, query languages)
- Don’t make a model more complex than needed
- Use subsets as convenient

Model Choices

More than one way to model reality

- Structural choices to make
- See later: Ontologies and Identities…

Alternative Relationships

- It is a best practice for ontologies to include inverse properties

```
ex:security foaf:isPrimaryTopicOf ex:Book1 .
ex:Book2 foaf:primaryTopic ex:security .
```

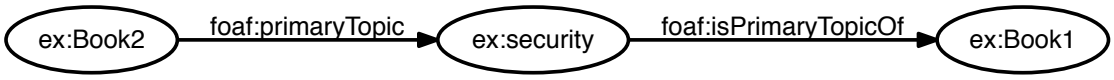


Figure 2: Bidirectionality

- Query on `foaf:primaryTopic` might miss half of data
  - Can infer inverse property based on ontology
  - Can expand query to deal with multiple cases

Alternative Structures

… But They Combine Well

Nice check of models

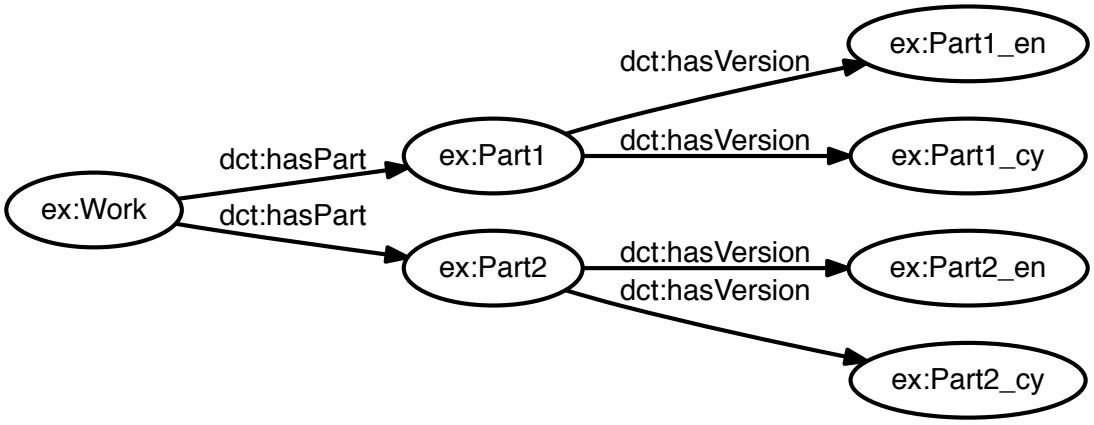


Figure 3: One option for structure

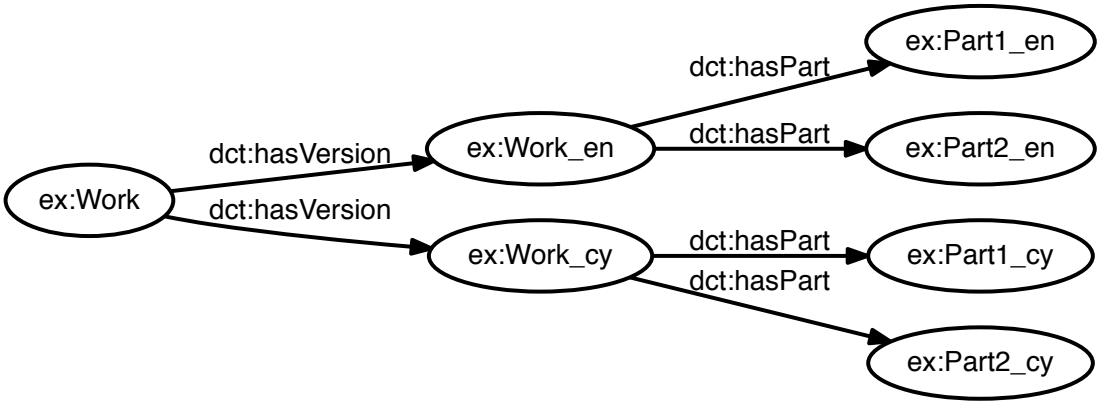


Figure 4: Another option for structure

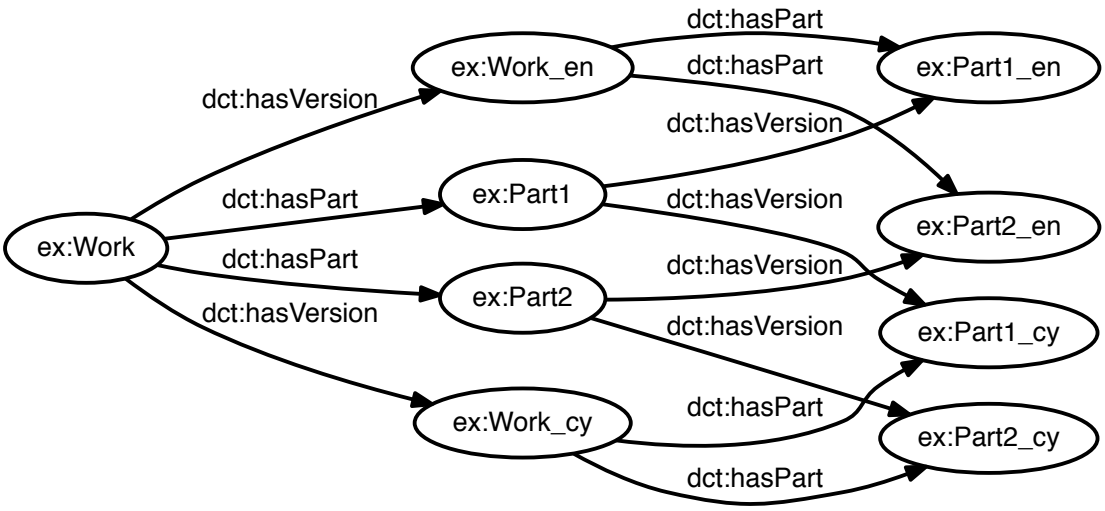


Figure 5: Both structure options combine OK

Alternatives - Mitigation

- **Develop and follow community practices**
- Not specific to linked data but linked data makes it obvious

---

The Open World

A Single Global Graph that everyone contributes to

- Great for data re-use
- Rich data from multiple sources
- Anyone can make assertions about anything
- Global identities
- Distributed: Can incrementally add to others descriptions
- Fits with the WWW: The Data Web

Technically: *If a statement is not asserted, then its truth-value is unknown, rather than false.*

- Data: Grass is Green.
- Question: Is Grass Red?

- Closed World: No
- Open World: I Don't Know

"Security Basics" by Alice

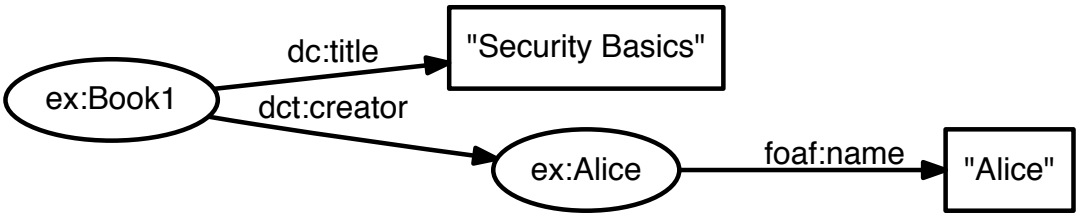


Figure 6: Book with title and author

In Turtle:

```
@prefix ex: <http://example.org/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dct <http://purl.org/dc/terms/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

ex:Book1 dc:title "Security Basics" .
ex:Book1 dct:creator ex:Alice .
ex:Alice foaf:name "Alice" .
```

"Security in Depth" by Alice

(no name for `ex:Alice`)

"Security Basics" and "Security in Depth"

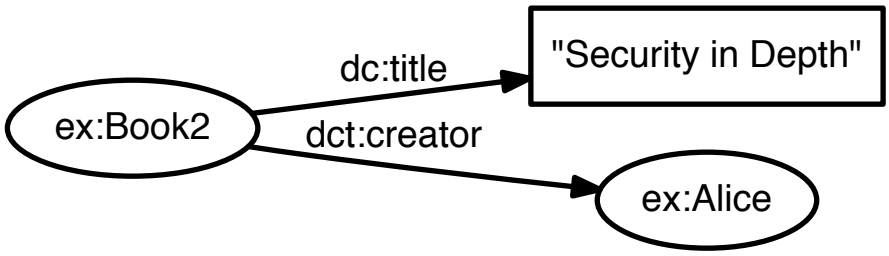


Figure 7: Book with title and author (note no name for Alice)

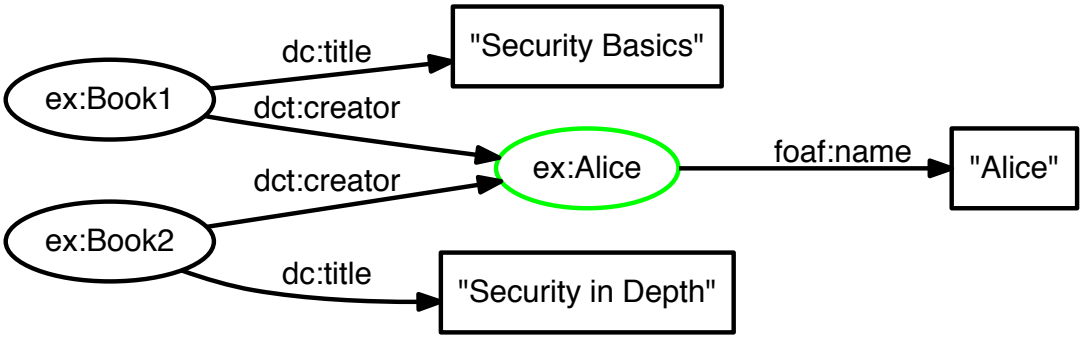


Figure 8: Books with shared author

“On Security” by Alice & Bob

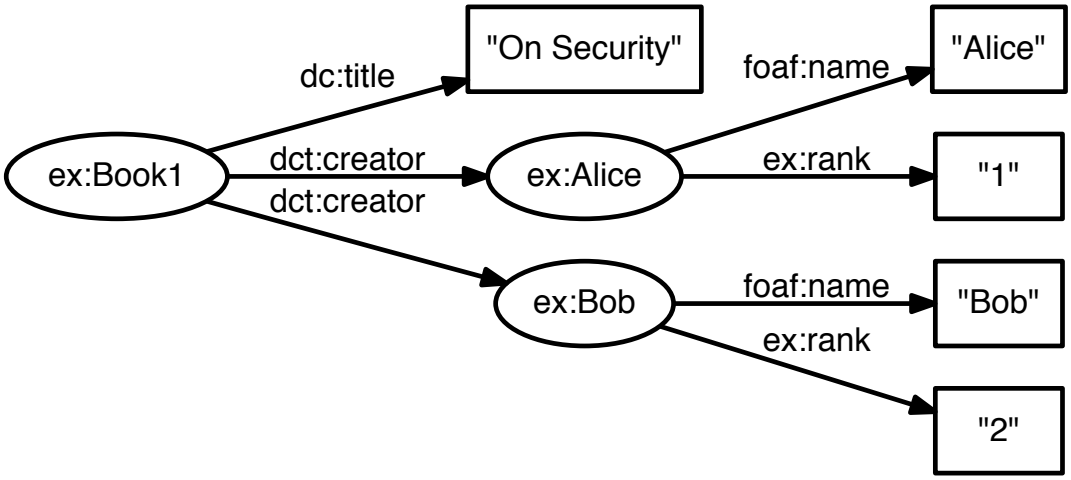


Figure 9: Book with title and two authors

“Tighter Security” by Bob & Alice

“On Security” and “Tighter Security”

Oops… garbage

Order, Lists and Sequences

Basic data structures for which RDF is *not* optimized

- Many different ways to model: simple list, `rdf:Seq`, `rdf:List`, using OAI-ORE proxies
- Beware Open World issues

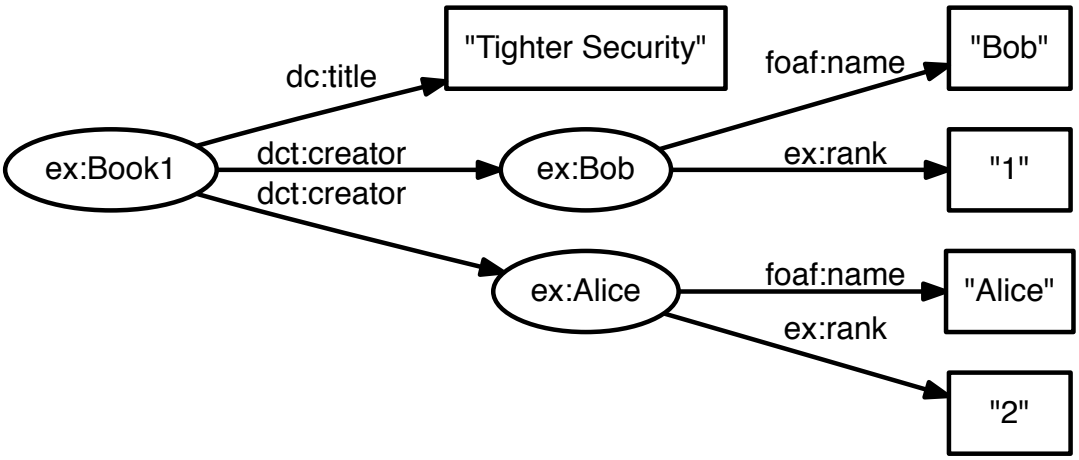


Figure 10: Another book with title and two authors

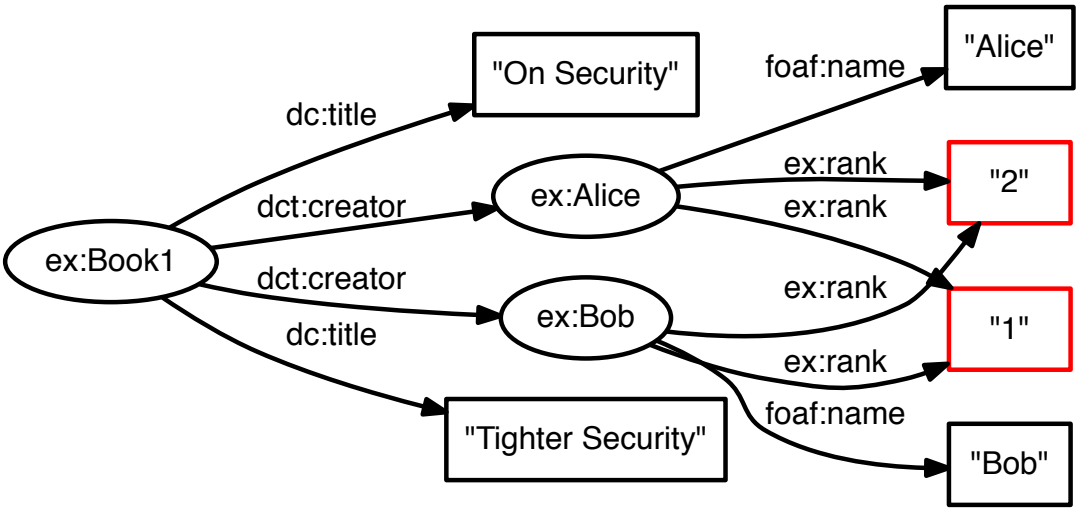


Figure 11: Two books with title and two authors but mangled author order information

Simple list

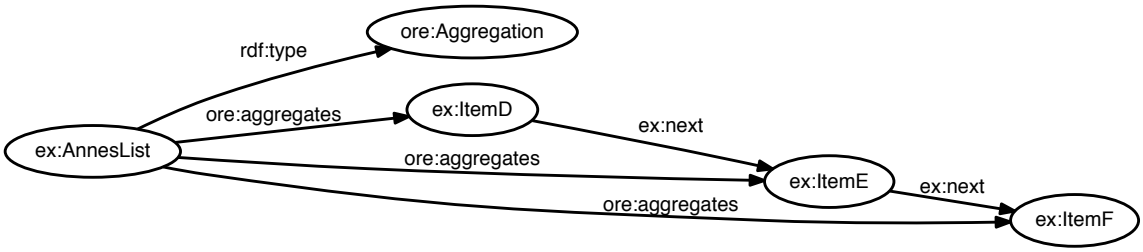


Figure 12: Simple list using ore:aggregates and ex:next for ordering

- Uniform `ore:aggregrates` for simple membership
- Order simply expressed by `ex:next`
- Not easy to query position in list (have to count)

Simple lists combined

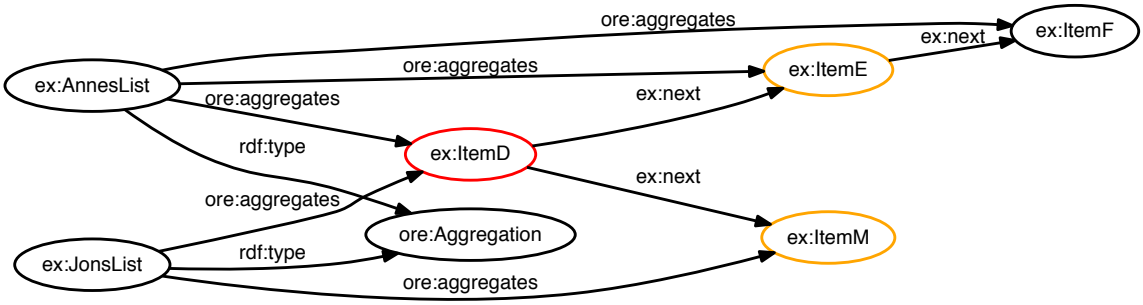


Figure 13: Simple lists combined showing order problem with ex:next outside local context

Oops, what is next from `ex:itemD`?

- `ex:next` works only in local context

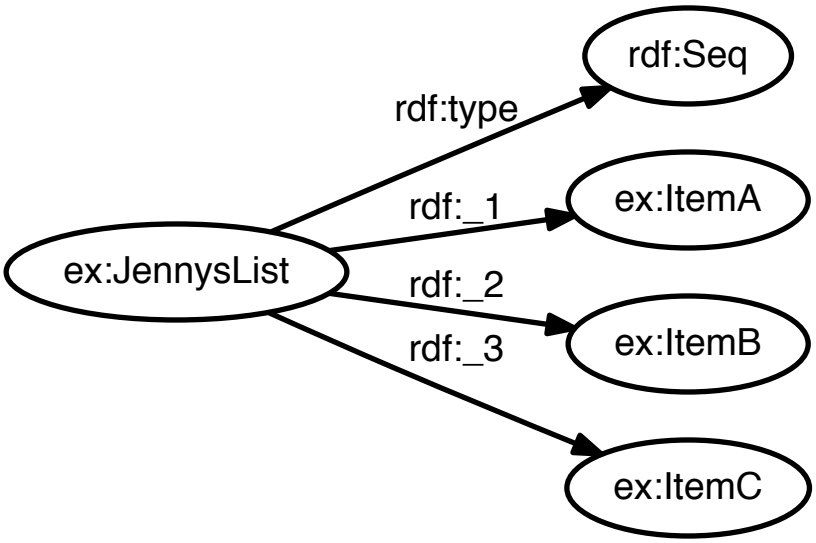


Figure 14: rdf:Seq

rdf:Seq

- Open ended
- Awkward properties to work with (`rdf:_1` etc.)
- No uniform member relation

---

rdf:List

- Closed
- Awkward to find members (especially as list grows)

Context with ORE proxy

- *Order the proxies, items OK in many lists*
- Useful beyond order, e.g. citation in context

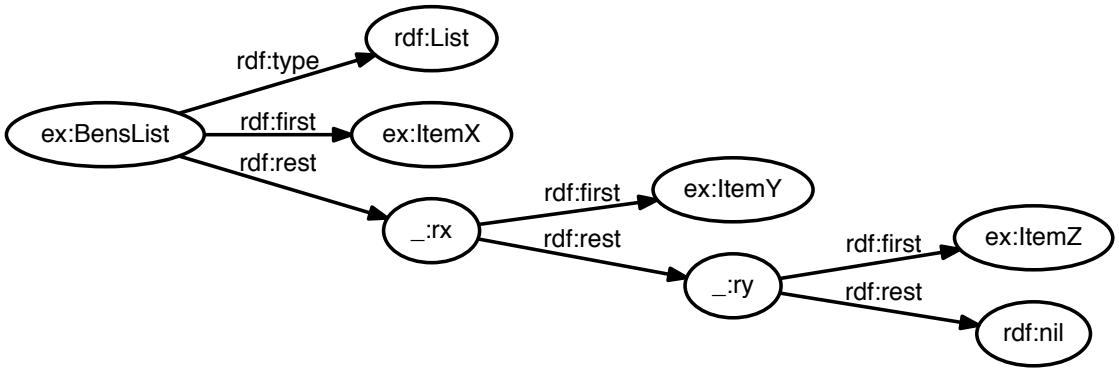


Figure 15: Ordering with rdf:List

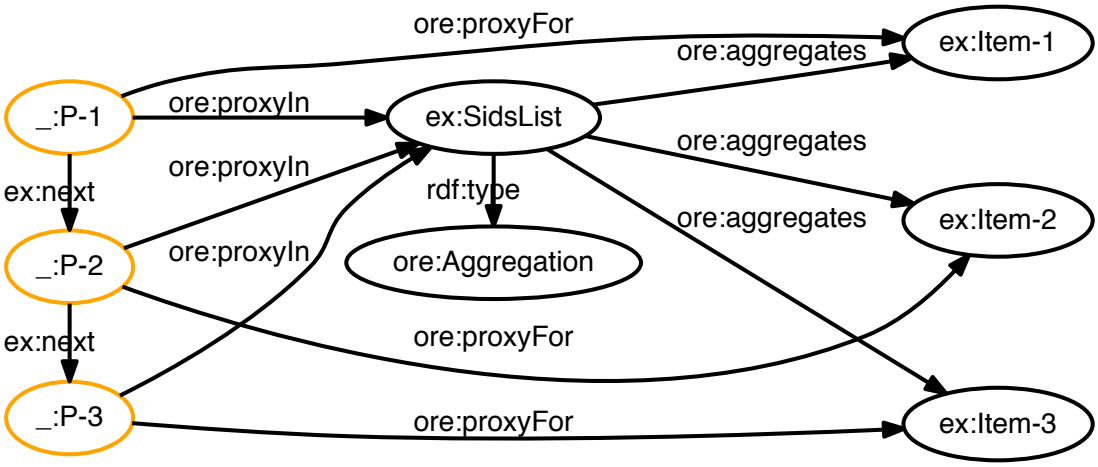


Figure 16: Ordering using ORE proxies

The Open World - Mitigation

- Local constructions, e.g. order, are complex.
  - Various serializations have support to hide complexity
- Remote assertions can change meaning
  - Trust is required in all data, not just linked data
  - Anyone can make assertions about anything *but you should understand who to listen to*
- *Local identity for local context* is good practice
- Harder to take short cuts, *forces understanding*
- Actually some grass is red

Ontologies and Identities

Shared ontologies increase interoperability

- Re-use of semantics makes it easier to build applications
- Need to understand terms, e.g. `dc:title` is 'name' or 'label', not a property title or Dr.
- Communities can develop own ontologies independently (as opposed to microdata/schema.org)

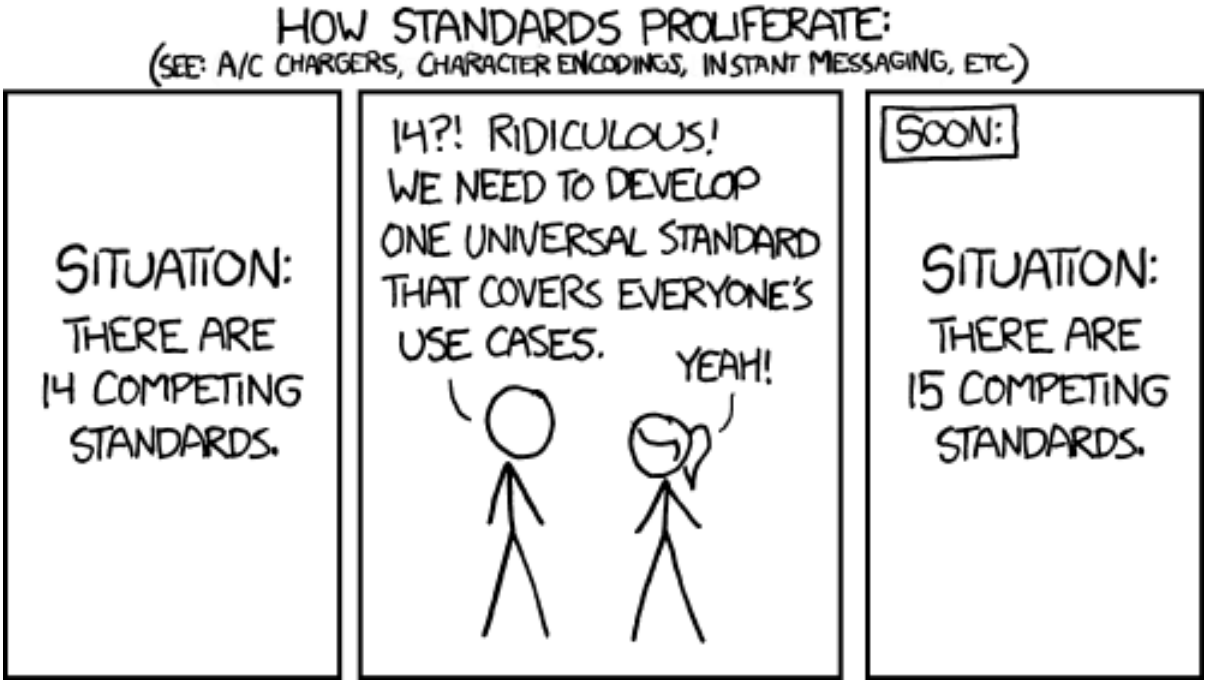
Shared identities make it possible for graphs to merge serendipitously

- Everyone can mint own IDs using http URIs
- By reusing ids, graphs will merge, creating new knowledge
- IDs have a maintenance overhead

Ontologies

> "The nice thing about … Ontologies … is that there's so many to choose from"

- Far far too many to choose from, hard to find the right one
- If almost right, do you *reuse and hope for the best, or specialize and create yet another ontology?*



http://xkcd.com/927/

Identities

> "The nice thing about … Identities … is that there's so many to choose from"

- Many schemes to choose from, which is the right one?
- As anyone can create identity for anything, they do
- Subtlety in whether identities equivalent
  - Identity often has a contextual component - does LANL's identifier for Oppenheimer differ from DBPedia's?
  - What are policies around maintenance?
- Avoid conflating identities

Ontologies and Identities - Mitigation

Use shared Ontologies and Identifiers wherever possible

- Can assert equivalences

  - Allows linking local to global
  - Assertions of equivalence are just assertions, same tooling
- There are well-known ontologies and identifier schemes
  - As use increases, winners will become (more) obvious

User Interfaces

Most user interfaces pretty bad anyway, but must be careful not to make linked data UIs worse

- How can existing data be leveraged to make entry easier?
- Do users need to know that they are dealing with linked data?
- Easy to have very specific queries which return few or no results
  - in web search, trend is toward giving approximate results by generalizing query

---

Serializations

- Too many serialization formats: N-Triples, N3, Turtle, TRIG, TRIX, RDF/XML, JSON-LD, NQuads,...
- Baseline RDF/XML is terrible:

  "RDF/XML was the Semantic Web's 3 Mile Island incident" – Manu Sporny

- Multiple formats means multiple identifiers for descriptions (one per format)
- Content Negotiation is a pain
- Not everyone implements every format = interop hell
- Leaves room for competing models/syntaxes

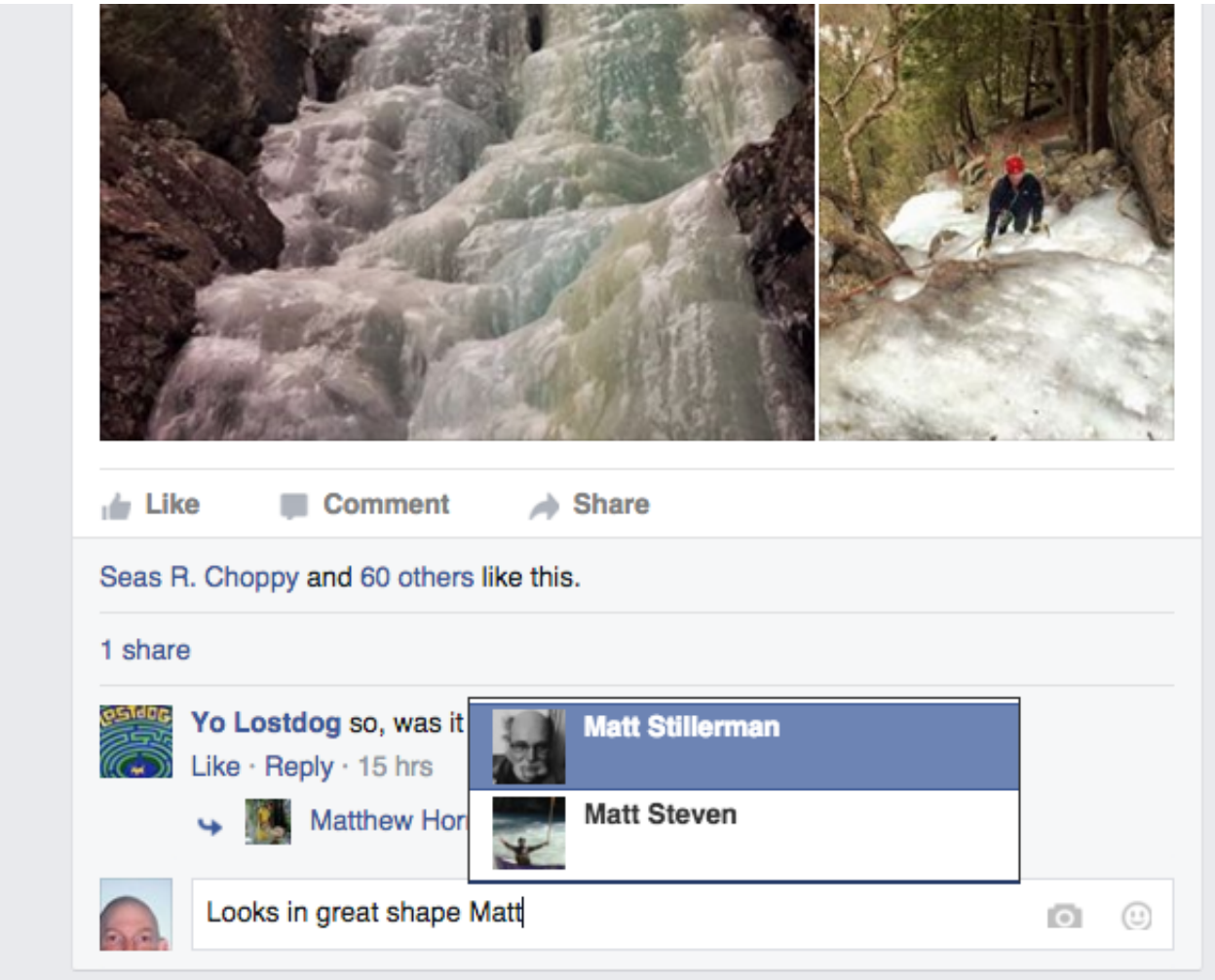


Figure 17: Facebook Autocomplete

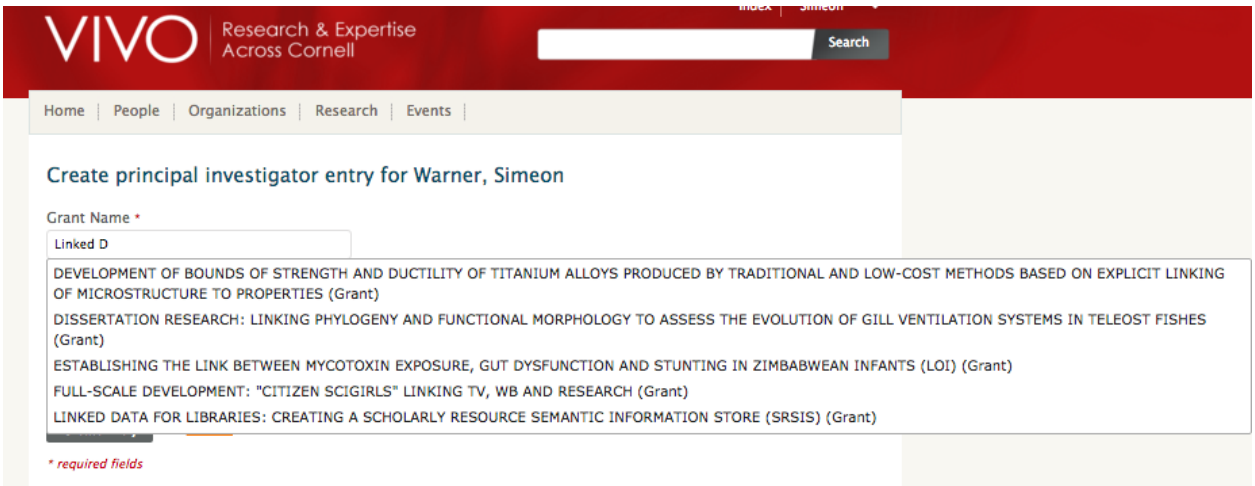


Figure 18: VIVO Autocomplete

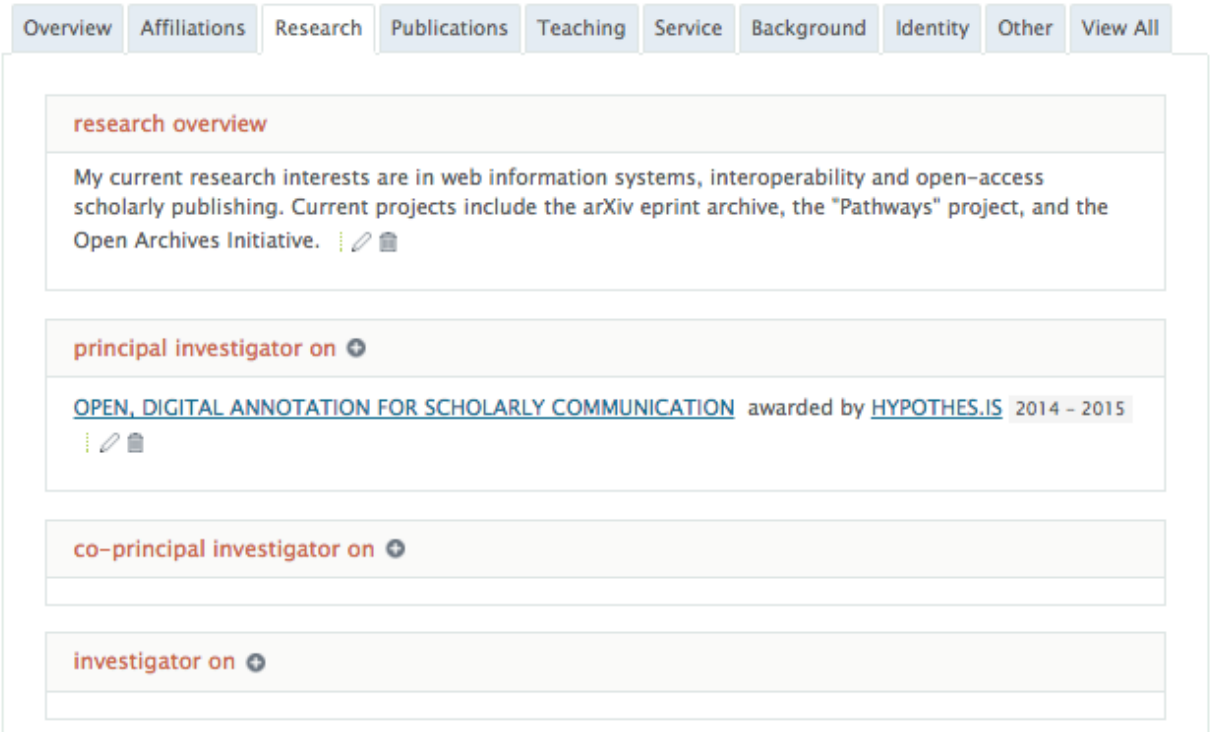


Figure 19: Parts of the VIVO UI show too much of the semantic web model

Figure 20: lodlive.it display just plain confusing

HTTP Range 14

If you don't already know, be happy that the title is gibberish. Instead, remember that we:

- Must have **separate URIs for resources and their descriptions**, we we get bad data:
  - URI-Homepage created 1996
  - URI-Simeon created before then
  - Confusion if URI-Homepage=URI-Simeon
- We can't return a Simeon over the web but we can use HTTP to direct users to the description
- **Follow established patterns, don't worry**

JSON-LD

New(ish) JSON-LD format is pretty good

```
{
  "@context":"http://iiif.io/api/presentation/2/context.json",
  "@id":"http://example.org/iiif/book1/annotation/anno1",
  "@type":"oa:Annotation",
  "motivation":"sc:painting",
  "resource":{
    "@id":"http://example.org/iiif/book1/res/p1.jpg",
    "@type":"dctypes:Image",
    "format":"image/jpeg"
  },
  "on":"http://www.example.org/iiif/book1/canvas/p1"
}
```

- Can use as plain JSON by ignoring `@context` etc.
- Used for IIIF, promoted in LDP

Diffs and change-sets

Without blank nodes, RDF diffs and change-sets are trivial lists of added and deleted triples - easy.

But, in general there are blank nodes so:

- *Diffs and change-sets very hard*
- No commonly accepted standard
  - hope with current work on W3C LD Patch
- Makes sync and incremental update hard
  - hope with LDP and protocols like ResourceSync

Serializations - Mitigation

Avoid RDF/XML, use JSON-LD and N-Triples/Turtle

- Look for good libraries
- Hope for community coalescence
  - JSON-LD adoption
  - LD Patch adoption

Technologies

- Pretty good tools for format conversions etc.
- Triplestores much less mature than relational and document databases
- SPARQL is standard query language, stable and widely implemented

Let's look at LD4L experience with 1 billion triples…

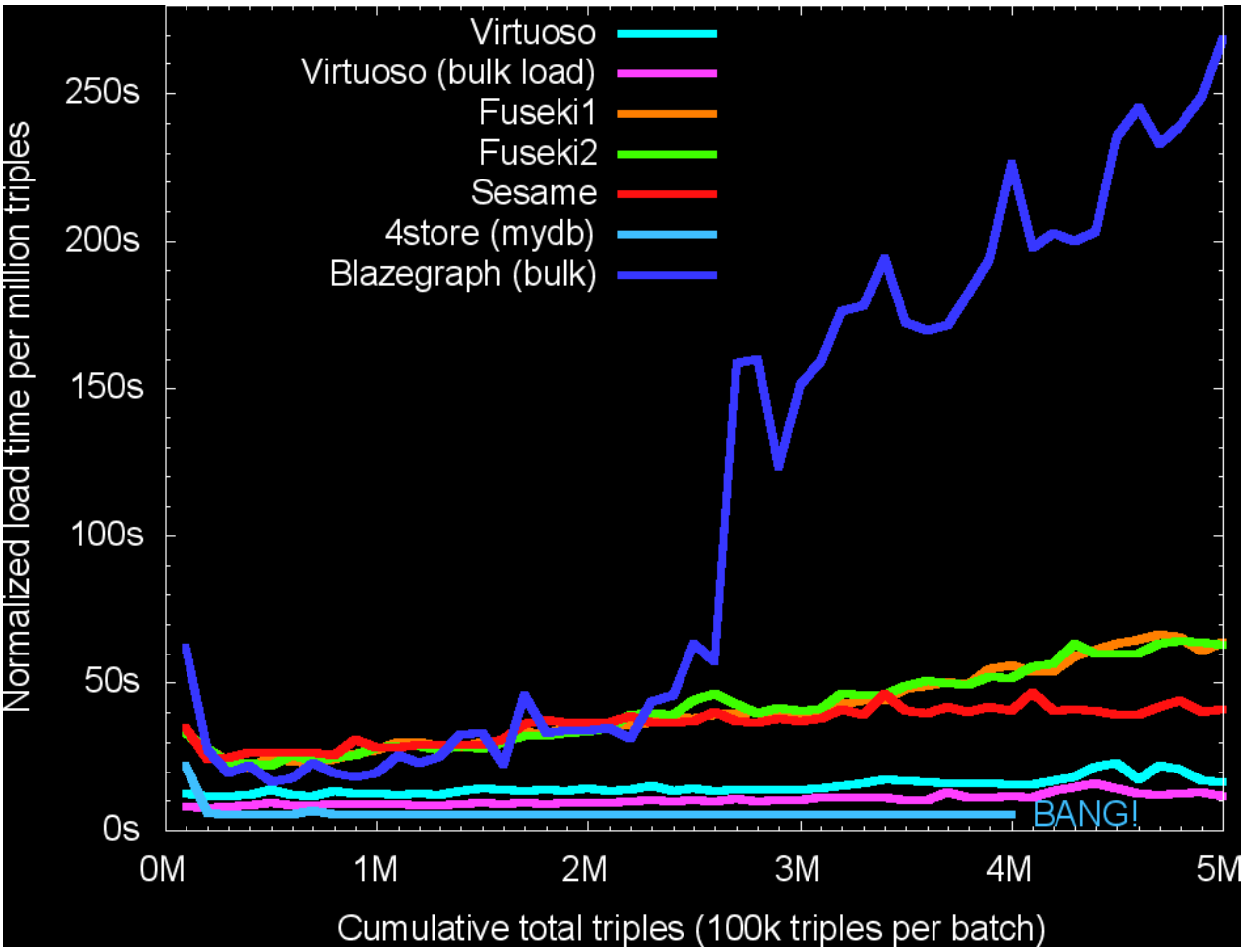


Figure 21: Initial load attempt, 5M triples. Several triplestore show terrible scaling. 4store failed part way due to problem on Mac OSX

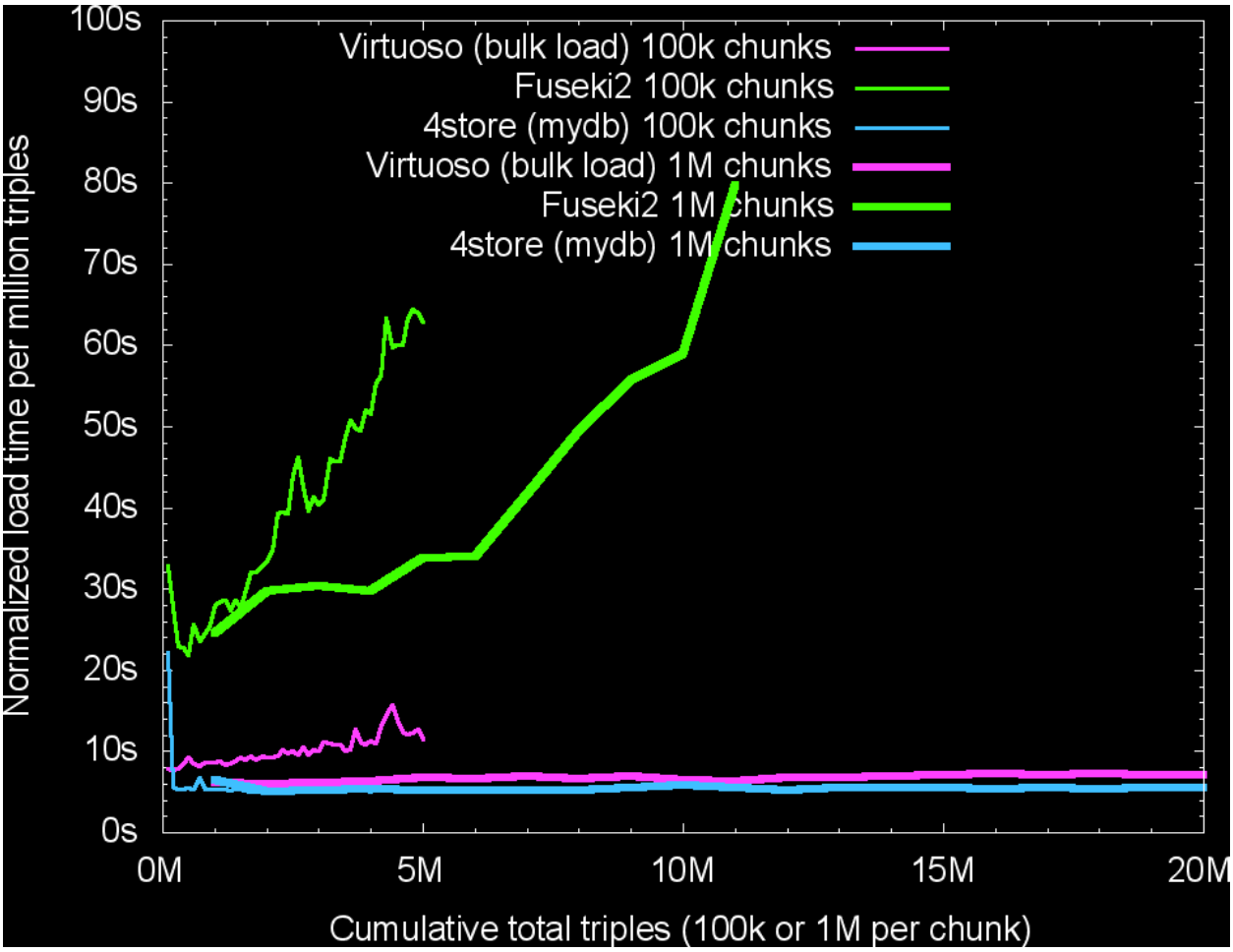


Figure 22: Second load attempt, 20M triples. Fuseki scaling bad, Virtuoso and 4store OK

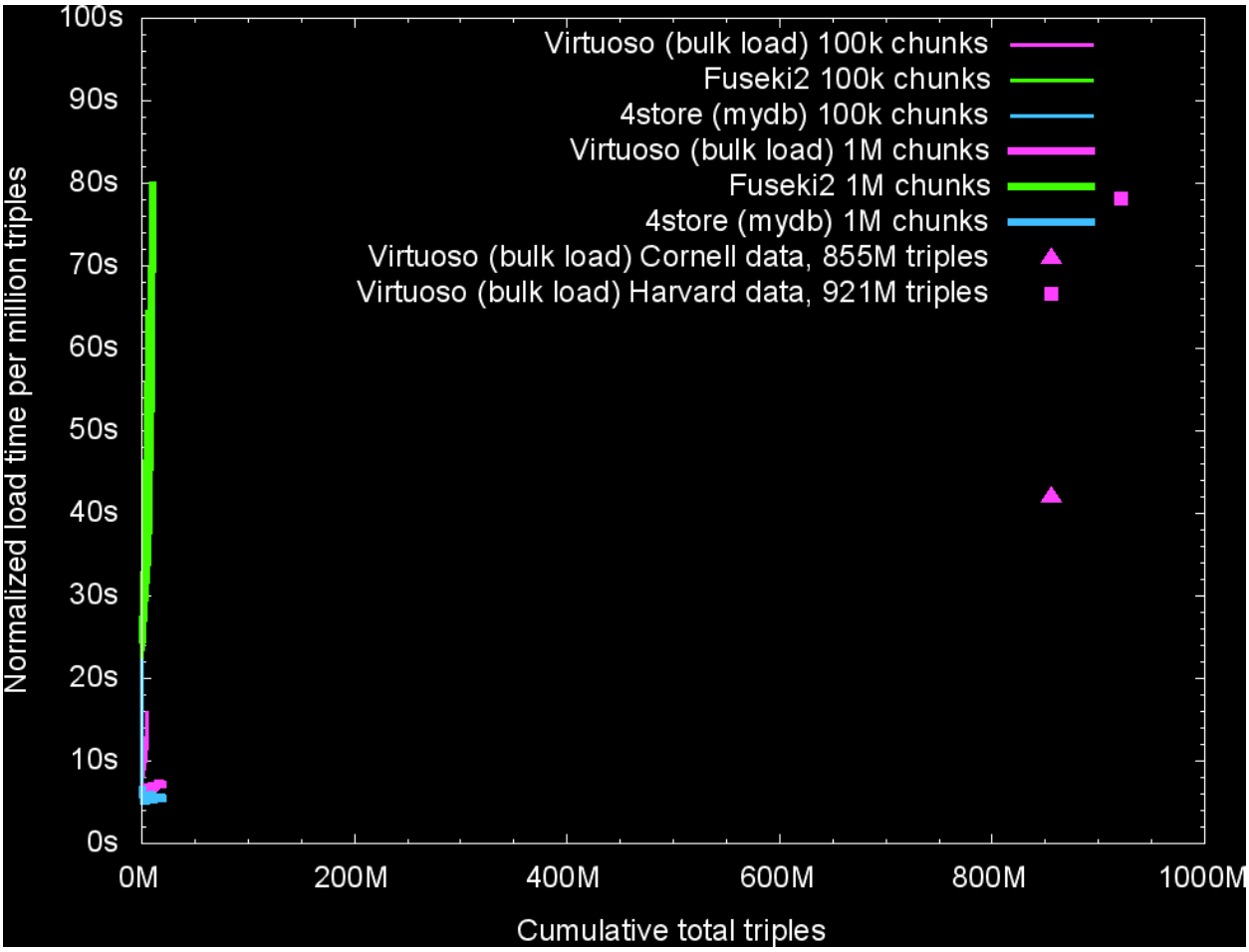


Figure 23: Successful loads of nearly 1 billion triples into Virtuoso showing load times around 1 minute per 1M triples

Technologies - Mitigation / Lessons

Major organizations now using Linked Data technologies at large scale

- Assign enough memory and compute power
- Consider commercial as well as open source software, still use open standards and data
- *We have more to learn, watch LD4L Labs…*

Temporality

> "The RDF data model is *atemporal*: RDF graphs are static snapshots of information" [RDF Concepts]

But…

- Resources change over time
  - Reality, Data and Ontologies
- Neither document nor data web has a solution
- Need to remain in sync in distributed environment

Temporality - Model it

- Ontologies to model (e.g Time Ontology in OWL)
- Various alternatives such as named graphs, reification, n-ary relationships (reified relations)
  - Structurally different - incompatible
  - All complex and/or verbose

Mitigation:

- Often needed only for parts of data model
- Consider other internal models even if RDF exposed

Temporality - Record it

Accept Linked Data as snapshot, save state of resources at various times and connect together sensibly.

- New URIs for every resource in every version doesn't work
- Coherency: does assertion still apply when other dataset changes?

Mitigation:

- Memento
- ResourceSync

Why Linked Data? How?

- Graphs offer more sophisticated modeling, and the complexity can be managed by tools and best practices
- Less "schema lock-in" and easier extensibility (both in open and proprietary systems)
- Open World needs care, but encourages good practices and enables distributed data to be combined
- Trust is required for reuse of any data, not just RDF
- Reuse ontologies and identities, work as community, foster interoperability
- Technologies still evolving, be nimble

That's all folks…